Discrete Applied Mathematics 36 (1992) 141-152
North-Holland

Indexing functions and time lower bounds for sorting on a mesh-connected computer

Yijie Han

Department of Computer Science, University of Kentucky, Lexington, KY 40506-0027, USA

Yoshihide Igarashi

Department of Computer Science, University of Kentucky, Lexington, KY 40506-0027, USA; and Department of Computer Science, Gunma University, Kiryu, 376 Japan

Miroslaw Truszczynski

Department of Computer Science, University of Kentucky, Lexington, KY 40506-0027, USA

Received 13 May 1988
Revised 6 June 1990

Abstract

Han, Y., Y. Igrashi and M. Truszczynski, Indexing functions and time lower bounds for sorting on a mesh-connected computer, Discrete Applied Mathematics 36 (1992) 141-152.

We introduce a parameter of indexing functions and show its relation to lower bounds for sorting algorithms on mesh-connected computers that follow from the Chain Theorem. We give lower and upper bounds for the parameter. Conclusions from our results are: (1) no matter what indexing function is used any sorting algorithm must execute $2.27n + \Theta(1)$ steps; (2) the best lower bound true for all indexing functions that we can hope to prove by the Chain Theorem argument is $2.5n + \Theta(1)$.

Keywords. Sorting, parallel computation, mesh-connected processor array, lower bound, complexity, indexing scheme, indexing function.

1. Introduction

In this paper we study a combinatorial problem that arises in considerations of sorting problems on a mesh-connected computer. As usual in such cases, it is of main concern to design fast algorithms and to prove lower bounds for the complexity of the problem to get an idea how good designed algorithms are.

Sorting on a mesh-connected computer has received much attention lately [1,2,4,5,7–16]. It turns out that the efficiency of a sorting algorithm depends on the indexing function used (see [1]), i.e., the function which for each i, $1 \le i \le n^2$, specifies the final location in the mesh of processors of the element of rank i. For a snake-like row-major indexing scheme an algorithm running in $3n + o(n)$ steps is known (Schnorr and Shamir [15]), and it is also known to be optimal (Kunde [4], Schnorr and Shamir [15]). So far, no sorting algorithm is known that would run in $(3-\varepsilon)n + o(n)$ steps, for some $\varepsilon > 0$. Also, the snake-like row-major and snake-like column-major indexing schemes are the only (up to trivial variations) indexing schemes for which fastest known algorithms are known to be optimal.

Clearly, sorting on an $n \times n$ mesh of processors must take at least $2n$ steps. The element whose final location is in the processor in a corner of the mesh may initially be stored in the processor in the opposite corner, and it takes at least $2n$ steps merely to move it to its proper final destination. This "structure-based" lower bound is too weak. No sorting algorithm running in $2n$ steps is known (and as we will see later, no such algorithm can exist). Only recently, a more powerful lower bound technique, known as the *joker-zone* method, was discovered by Kunde [4] and Schnorr and Shamir [15]. They used the method to show that $3n$ is a lower bound for the running time of any algorithm sorting into snake-like row-major or row-major indexing schemes. Their method was subsequently refined by Han and Igarashi [1]. They developed an argument based on the so-called *Chain Theorem*, and proved that $(1 + \sqrt{6}/2)n + \Theta(1)$ is a lower bound for the running time of any sorting algorithm, no matter what indexing function is used. (In the paper we use the following convention: for functions f, g and h defined on the same set D

(1) $f(x) = g(x) + \Theta(h(x))$ means that there are constants A and B such that

$$g(x) - Ah(x) - B \le f(x) \le g(x) + Ah(x) + B;$$

(2) $f(x) \ge g(x) + \Theta(h(x))$ means that there are constants A and B such that

$$f(x) \ge g(x) + Ah(x) + B;$$

(3) $f(x) \le g(x) + \Theta(h(x))$ means that there are constants A and B such that

$$f(x) \le g(x) + Ah(x) + B.)$$

Han and Igarashi [1] also constructed an example of a poor indexing scheme; any algorithm sorting into this indexing scheme must execute at least $4n + \Theta(\sqrt{n})$ steps. Kunde [6] applied a joker-zone argument to derive an indexing scheme independent lower bound of $2.25n$ and suggested that this bound may be the limit of the joker-zone argument. Several of the results discussed above were extended to the case of meshes with wrap-around connections [6] (in particular, an indexing scheme independent lower bound of $1.5n$ was given there) and d-dimensional mesh-connected computer [2,5].

The main contribution of this paper is the formalization of the Chain Theorem of Han and Igarashi [1] and an in-depth study of the power of the theorem. To this end, for an indexing function I we define a combinatorial parameter called *stretch* and denoted $s(I)$, and we show that lower bounds implied by the Chain Theorem directly depend on this parameter. This new version of the Chain Theorem is used in the subsequent sections to obtain two main results of the paper. Our first result (Section 3) provides a lower bound for $s(I)$; this allows to prove that independent of an indexing function, every sorting algorithm requires at least $2.27n$ steps, an improvement over the old bounds $(1+\sqrt{6}/2)n+\Theta(1)$ of [1] and $2.25n$ of [6]. Our second result (Section 4) exhibits an indexing function I with $s(I)=0.5n+\Theta(1)$. This outlines limits for the power of the Chain Theorem. More precisely, it says that the best universal (independent of an indexing function) lower bound we can hope to obtain by an argument based exclusively on the Chain Theorem is $2.5n$.

2. Preliminaries and problem formulation

We consider a general model of a synchronous $n \times n$ mesh-connected processor array as given in [15]. It is denoted by $M(0..m, 0..m)$; here, and throughout the paper $m=n-1$. Each processor at location (i,j), $0 \le i, j \le m$, is denoted by $M(i,j)$. The distance between $M(i_1,i_2)$ and $M(j_1,j_2)$ is defined as $|i_1-j_1|+|i_2-j_2|$ and denoted by $d((i_1,i_2),(j_1,j_2))$. Processor $M(i_1,i_2)$ is directly connected with processor $M(j_1,j_2)$ if and only if $d((i_1,i_2),(j_1,j_2))=1$. This model is illustrated in Fig. 1. All n^2 processors work in parallel with a single clock, but they may run different programs. As for sorting computation, the initial contents of $M(0..m, 0..m)$ are assumed to be n^2 items drawn from a totally ordered set, where each processor has exactly one item. The final contents of $M(0..m, 0..m)$ is the sorted sequence of the items in a specific order. In one step each processor can communicate with all of its directly connected neighbor processors. The interchange of items in a pair of directly connected processors or the replacement of the item in a processor with the item in one of its directly connected processors can be done in one step. The computing time is defined as the number of parallel steps of such basic operations to reach the final configuration. The biggest distance between two processors in the same row (or column) is m.

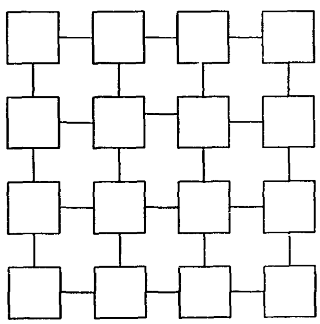

Fig. 1. A mesh-connected processor array.

A one-to-one function $I: \{0, 1, \ldots, m\}^2 \to \{1, 2, \ldots, n^2\}$ is called an *indexing function*. Given an indexing function I, the goal is to sort n^2 items initially stored in the n^2 processors so that when the algorithm terminates, the item of rank k (the kth smallest) is located in processor $M(i,j)$, where $I(i,j) = k$. A family of indexing functions, one for each n, sharing some property is called an *indexing scheme*. Various indexing schemes are shown in Fig. 2.

A subset of $M(0..m, 0..m)$ is called a *region*. For a region S the number of processors in S will be called the *cardinality* of S and will be denoted $|S|$. In the sequel we often assume that the mesh of processors is embedded in the real plane in such a way that processor $M(i,j)$ is being located in point (i,j). In this geometric setting the cardinality of a region is easy to compute. If P is a convex polygon in the plane, then the set of processors located in points of P has cardinality equal to $|P| + \Theta(p)$, where $|P|$ is the area of P and p is the perimeter of P. The formula remains true if P is not convex but is the union of two interior-disjoint convex polygons.

Throughout the paper, for any two real numbers a and b, $[a,b]$ denotes the set of all integers j, such that $a \le j \le b$. Each set of processor locations of the form $I^{-1}([a,b])$ is called a *chain under indexing function I* (or a *chain* if I is understood). The *length* of such a chain is $|[a,b]|$. If (i_1, i_2) is in $\{0, m\}^2$ and x is a positive real number, $\{M(j_1, j_2)\colon d((i_1, i_2), (j_1, j_2)) \le x\}$ is called a *corner* region and is denoted

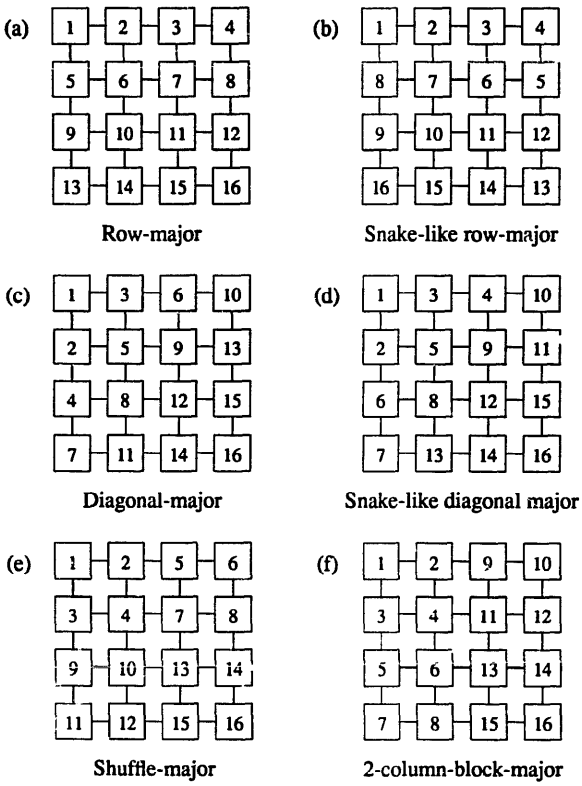

Fig. 2. Various indexing schemes.

by CREG$((i_1,i_2);x)$. An *open* corner region is the set $\{M(j_1,j_2)\colon d((i_1,i_2),(j_1,j_2))<x\}$, and it is denoted by CREG$_O((i_1,i_2);x)$. The set of all processors that are at distance at least $m-x$ from all four processors $M(0,0)$, $M(0,m)$, $M(m,0)$, and $M(m,m)$ is called a *center* region and is denoted by CENT(x).

Consider now an indexing function I and a corner region $R=$ CREG$((i,j);x)$, for some real x, $0\le x\le 2m$. Let c be the length of a longest chain contained in R, and let $t(R)$ be the smallest real number t such that $c\le |\text{CREG}((i,j);t)|$ ($t(R)$ is well defined, in fact, it is an integer). Finally, put $s(R)=x-t(R)$. The *stretch* $s(I)$ of I is defined as $s(I)=\sup s(R)$, where the supremum is taken over all corner regions R.

The following theorem has been derived in [1] and is called the Chain Theorem. We present a proof of this theorem here since the theorem is given in a new formulation, in terms of the parameter $s(I)$. The theorem gives a lower bound for sorting into the order specified by an indexing function I in terms of $s(I)$ and is easier to apply than the previous version.

Theorem 2.1. *Let I be an indexing function. Then, every algorithm for sorting n^2 items into the order specified by I takes at least $2n+s(I)+\Theta(1)$ steps.*

Proof. Let us consider a region $R=$ CREG$((i,j);x)$, for some $i,j\in\{0,m\}$ and some real number x, $0\le x\le 2m$. Let c, $t(R)$ and $s(R)$ be defined as above. Consider any sorting algorithm A. After $2n-t(R)+\Theta(1)$ steps of this algorithm, the current content y of the processor $M(i,j)$ does not depend on the values initially stored in the processors of the corner region $R_1=$ CREG$((m-i,m-j),t(R))$ (by the routine distance argument). By setting appropriately the values of the elements initially stored in the processors of region R_1 we can force y to have any of the $c+1$ consecutive ranks in the whole set. One of these ranks must be assigned by I to a location outside region R (recall that the length of the longest chain in R is c). Thus, in the worst case, algorithm A has still to perform at least $x+\Theta(1)$ steps. Consequently, the total number of steps required is at least $2n-t(R)+x+\Theta(1)=2n+s(R)+\Theta(1)$. Since R has been chosen arbitrarily, the assertion follows. □

This theorem points to the importance of the parameter $s(I)$ in studying lower bounds for sorting on mesh-connected computers. In this paper we study the parameter $s_n=\min s(I)$, where the minimum is taken over all possible indexing functions on an $n\times n$ mesh of processors. We show that $0.27n\le s_n$ (hence, every sorting algorithm must require at least $2.27n$ steps), and that $s_n\le 0.5n$ (hence, the best universal lower bound that can be obtained using the Chain Theorem only is $2.5n$).

3. Lower bound

In this section we will show two theorems each giving a lower bound for s_n. The first one gives the lower bound initially presented in [1]. The new form of the Chain Theorem allows for an especially elegant and short proof. We present it here to illustrate the way the Chain Theorem is applied and to help better understand the approach behind the proof of the improved lower bound for s_n which is the main result of this section (Theorem 3.3). We start with the following lemma.

Lemma 3.1. *Let a be a real number, $0 \le a \le \frac{1}{2}$.*

(a) $I^{-1}([an^2,(1-a)n^2]) \subseteq \mathrm{CENT}((1-\sqrt{2a})n+s(I)+\Theta(1))$.

(b) *Let $x_a = \inf\{x\colon\ |\mathrm{CENT}(x)| \ge |[an^2,(1-a)n^2]|\}$. We have*

$$x_a = \begin{cases} n\sqrt{1/2-a}+\Theta(1), & \text{if } 0 \le a \le \frac{1}{4},\\ n(1-\sqrt{a})+\Theta(1), & \text{if } \frac{1}{4} < a \le \frac{1}{2}.\end{cases}$$

Proof. (a) Let b be an integer, $b \in [an^2,(1-a)n^2]$. Suppose $b \notin \mathrm{CENT}((1-\sqrt{2a})n+s(I)+1)$. Then, for some $(i,j) \in \{0,m\}^2$, $b \in \mathrm{CREG_O}((i,j);n\sqrt{2a}-s(I)-2)$. Let $d((i,j),I^{-1}(b))=x$ and let $R=\mathrm{REG}((i,j),2m-x)$. Clearly, $x < n\sqrt{2a}-s(I)-2$, and the longest chain contained in R has length at most $(1-a)n^2$. Hence, $t(R) \le 2n-n\sqrt{2a}$. Consequently, $s(R)=2m-x-t(R)>s(I)$, a contradiction.

(b) Follows directly from the following formuia for the number of elements in a center region:

$$|\mathrm{CENT}(x)| \le \begin{cases} 2x^2+\Theta(x), & \text{if } 0 \le x \le \frac{1}{2}m,\\ n^2-2(m-x)^2+\Theta(m-x), & \text{if } \frac{1}{2}m < x \le m. \end{cases} \qquad \square$$

Theorem 3.2 (Han and Igarashi [1]). $s_n \ge (\sqrt{6}/2-1)n+\Theta(1)$.

Proof. Let us consider an arbitrary indexing function I. Under the notation from Lemma 3.1 we have

$$x_a \le n+s(I)-\sqrt{2a}n+\Theta(1)$$

(this follows from Lemma 3.1(a)). Hence (by Lemma 3.1(b)),

$$s(I) \ge \begin{cases} n(\sqrt{1/2-a}+\sqrt{2a}-1)+\Theta(1), & \text{if } 0 \le a \le \frac{1}{4},\\ n(\sqrt{2a}-\sqrt{a})+\Theta(1), & \frac{1}{4} < a \le \frac{1}{2}. \end{cases}$$

Maximizing the right-hand side with respect to a we get $s(I) \ge (\frac{1}{2}\sqrt{6}-1)n+\Theta(1)$, as claimed. $\square$

Next, we present an improvement on this result. We first prove an auxiliary lemma.

Lemma 3.3. *Consider two center regions B_1 and B_2, and regions C_i and D_i, $i=1,2,3,4$, as shown in Fig. 3. Define $H_i=C_i \cup D_i \cup C_{i+1}$, $i=1,2,3$, and $H_4=C_1 \cup D_4 \cup C_4$. Put $B=B_1-B_2$, $b=|B|$, and $d=|D_1|$ (regions D_i have all the same*

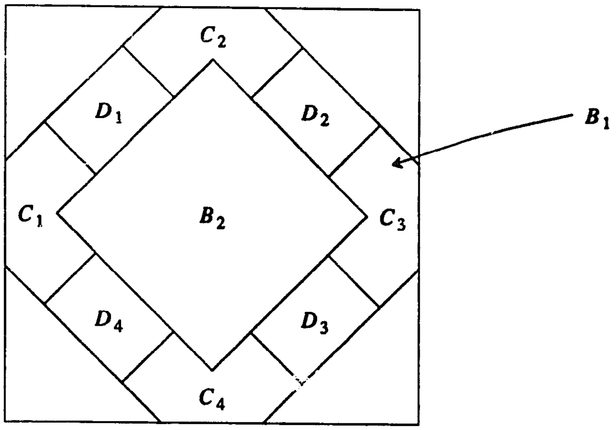

Fig. 3.

size). Assume that more than $\frac{1}{2}b + 2d$ elements of B are colored with blue and green and suppose that there is at least one element of each color. Then at least one of the regions H_i contains elements of both colors.

Proof. Without loss of generality we may assume that there are no less blue elements than green elements.

Case 1: *There exists a green element in* $\bigcup_{i=1}^{4} C_i$. Without loss of generality we may assume that there is a green element in C_1. Since more than $\frac{1}{4}b + d$ elements are colored blue, there exists a blue element not in $C_3 \cup D_2 \cup D_3$, and the assertion of the lemma holds.

Case 2: *All green elements are in* $\bigcup_{i=1}^{4} D_i$. Without loss of generality we may assume that there is a green point in D_1. Since more than $\frac{1}{2}b + 2d$ elements in B are colored and no green element is in $C_1 \cup C_2$, there is at least one blue element in $C_1 \cup C_2$. Thus, the assertion of the lemma holds in this case, too. □

Theorem 3.4. $s_n \geq 0.27n + \Theta(1)$.

Proof. Let I be an arbitrary indexing function. Suppose that $s(I) < 0.27n$. Consider two sets $A_1 = I^{-1}([0.21n^2, 0.79n^2])$ and $A_2 = I^{-1}([0.395n^2, 0.605n^2])$. By Lemma 3.1(a), for every sufficiently large n, $A_i \subseteq B_i$, where $B_i = \mathrm{CENT}(x_i m)$, $x_1 = 0.622$ and $x_2 = 0.3812$ (recall that $m = n - 1$). Regions B_i and other regions we will consider in the proof are shown in Fig. 4. Let $B = B_1 - B_2$ and $l = 0.1123024$. Color all elements in $I^{-1}([0.21n^2, 0.395n^2])$ in blue and all elements in $I^{-1}([0.605n^2, 0.79n^2])$ in green. Altogether, there are $0.37n^2 + \Theta(1)$ colored elements. These colored elements must be located in B_1. At most $|B_2| - (0.21n^2) + \Theta(1)$ of them can be located in B_2, as $A_2 \subseteq B_2$ and $|A_2| = 0.21n^2 + \Theta(1)$. Since $|B_2| = 2(x_2 m)^2 + \Theta(m) = 2(x_2 n)^2 + \Theta(n)$, at least $0.2893n^2 + \Theta(n)$ of the colored elements are located in B. In particular, it follows that B contains both blue and green points. Observe that $|B| = 2(x_1 m)^2 -$

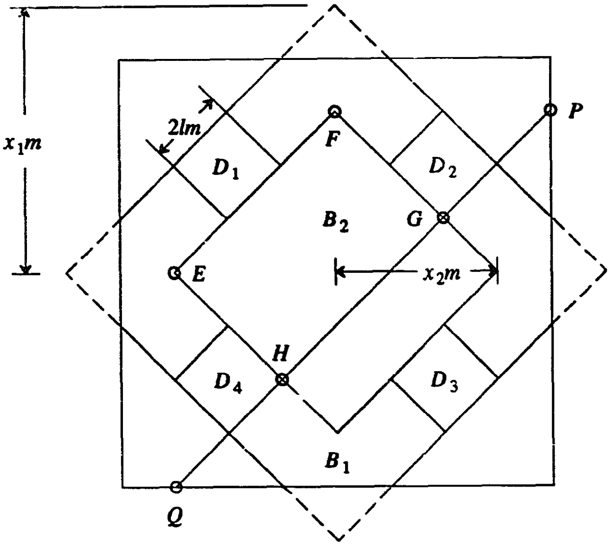

Fig. 4.

$2(x_2m)^2 - 4(x_1m - \frac{1}{2}m)^2 + \Theta(m)$. Denote by d the common cardinality of regions D_i and observe that $d = \sqrt{2}l(x_1 - x_2)m^2 + \Theta(m)$. Hence, the number of colored elements in B is bigger (for sufficiently large n) than $\frac{1}{2}|B| + 2d$. Thus, by Lemma 3.3, there are both blue and green points in one of the regions H_i (see the notation of Lemma 3.3), say in H_3. Consider now set $A_2 = I^{-1}([0.395n^2, 0.605n^2])$. It has $0.21n^2 + \Theta(1)$ elements. All of them belong to B_2. Notice, that the cardinality of the region $EFGH$ is given by $(x_2m)^2 + \sqrt{2}lx_2m^2 + \Theta(m)$ and thus it contains at most $0.206n^2 + \Theta(n)$ elements. Therefore, for every sufficiently large n, there is an element in A_2 that belongs to H_3. Let R be the corner region determined by the line PQ and containing H_1. It follows that the longest chain in R has length at most $0.395n^2 + \Theta(1)$. Thus, $s(R) \geq 0.27n + \Theta(1)$, as required. □

Remark. The values for x_1, x_2 and l where found by maximizing formulas similar to the one that appears in the proof of Theorem 3.1. Since in the case of the proof of Theorem 3.3 the formulas used (and their derivations) are much more complicated, we decided not to include them into the proof and only use the values x_1, x_2 and l these formulas imply.

We conclude this section with a theorem being a corollary of Theorems 2.1 and 3.4.

Theorem 3.5. *No matter what indexing function is used, any algorithm for sorting n^2 items on a mesh-connected computer takes at least $2.27n + \Theta(1)$ steps.*

4. Limit of the chain argument

The key element of the arguments of the preceding section is the Chain Theorem. In this section we study the power of the chain argument. It turns out that the best lower bound we can hope to obtain using this type of argument is $2.5n+\Theta(1)$. To justify this claim we will construct an indexing function I with $s(I)\leq 0.5n+\Theta(1)$.

Before we define a suitable indexing function let us note that the sets $\mathrm{CREG_O}((i,j),\lceil\frac{1}{2}m\rceil)$, for $i,j\in\{0,m\}^2$ and $\mathrm{CENT}(\lceil\frac{1}{2}m\rceil)$ form a partition of the set of all processors. For brevity, we denote $\mathrm{CREG_O}((i,j),\lceil\frac{1}{2}m\rceil)$, for $i,j\in\{0,m\}^2$ by $A_{i,j}$ and $\mathrm{CENT}(\lceil\frac{1}{2}m\rceil)$ by C. Let $a=|A_{i,j}|$ (it does not depend on i and j) and $c=|C|$. Let us assume now that an indexing function I satisfies the following requirements:

(1) Processors in $A_{0,0}$ (respectively $A_{0,m}$) will be assigned odd (respectively even) integers from $\{1,\ldots,2a\}$, processors in C will be assigned elements from $\{2a+1,2a+2,\ldots,n^2-2a\}$, and processors in $A_{m,0}$ (respectively $A_{m,m}$) will be assigned odd (respectively even) integers from $\{n^2-2a+1,n^2-2a+2,\ldots,n^2\}$.

(2) For every $x=M(i_1,j_1)$ and $y=M(i_2,j_2)$,

(a) if x and y are both in $A_{0,0}$ or in $A_{m,m}$ and $i_1-j_1<i_2-j_2$, then $I(x)>I(y)$,

(b) if x and y are both in $A_{0,m}$ or in $A_{m,0}$ and $i_1+j_1<i_2+j_2$, then $I(x)>I(y)$,

(c) if x and y are both in C and $i_1<i_2$, then $I(x)<I(y)$.

An example of an indexing function satisfying requirements (1) and (2) (for $n=9$) is given in Fig. 5. It is clear that indexing functions satisfying (1) and (2) exist for every positive n. (Top leftmost corner contains $M(0,0)$, top rightmost corner contains $M(0,m)$.)

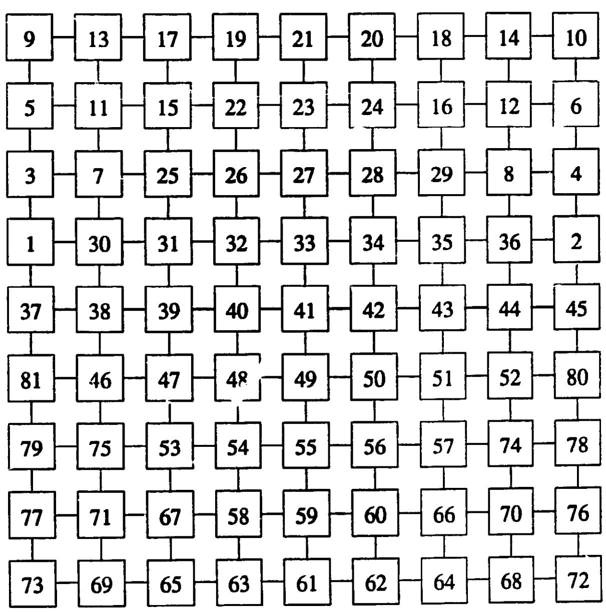

Fig. 5.

Theorem 4.1. *If an indexing function satisfies requirements* (1) *and* (2), *then* $s(I)=0.5n+\Theta(1)$. *Hence*, $s_n \le 0.5n+\Theta(1)$.

Proof. To prove the theorem, we show that no matter what corner region $R=\mathrm{CREG}(i,j;x)$ is used, $s(R)=x-t(R)\le 0.5n+\Theta(1)$. We consider first the case when $i=m$ and $j=m$, and we split it into four subcases according to x. In cases 2, 3 and 4, the elements contained in the interior of region B indicated with the bold line in Fig. 6(a), (b) and (c), respectively, form a chain. (In this figure, we assume that the top leftmost corner contains processors $M(0,0)$ and the top rightmost corner contains processor $M(0,m)$.) The length of this chain is equal to $|B|+\Theta(p)$, where $|B|$ is the area of the polygon B and p is the perimeter of B. (Note that in (c) B is the union of two interior-disjoint convex polygons.)

Case 1: $0\le x\le 0.5m$. In this case $s(R)\le x\le 0.5m$, as required.

Case 2: $0.5m<x\le m$. In this case, (see Fig. 6(a)) $|B|=\frac{3}{4}b^2$ and $p=\Theta(b)$, where $b=x-0.5$. Hence, $t(R)=b\sqrt{3/2}+\Theta(1)$. Since $0<b\le 0.5m$, $s(R)=x-t(R)\le 0.5m+\Theta(1)$.

Case 3: $m<x\le 1.5m$. In this case, (see Fig. 6(b)) $|B|=\frac{1}{4}(2m^2-2bm-b^2)$ and $p=\Theta(m)$, where $b=1.5m-x$. Since $0.1875m^2+\Theta(m)\le |B|\le 0.5m^2$, $s(R)=1.5m-b-\sqrt{(2m^2-2bm-b^2)/2}+\Theta(1)$. As $0\le b<0.5m$, also in this case we have $s(R)\le 0.5m+\Theta(1)$.

Case 4: $1.5m<x\le 2m$. $|B|=0.75m^2+\frac{1}{2}(x-1.5m)^2$ and $p=\Theta(m)$. $|B|\ge 0.75m^2+\Theta(m)$ so, $s(R)=\sqrt{0.5m^2-(x-1.5m)^2}+x-2m+\Theta(1)$. Since, $1.5m<x\le 2m$, $s(R)\le 0.5m+\Theta(1)$ follows.

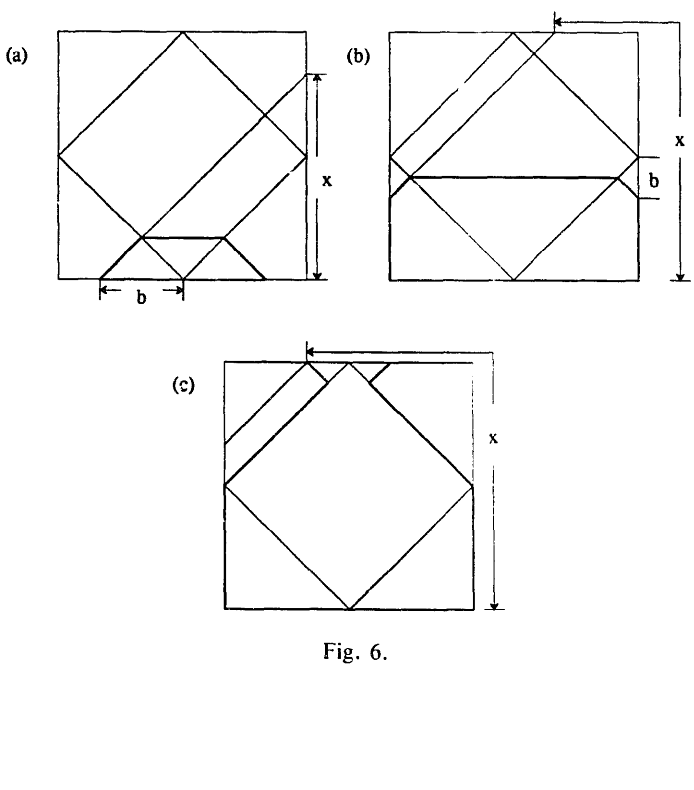

Fig. 6.

In the other three cases for the corner we consider identical subcases and get the same formulas for $t(R)$ and $s(R)$ as in the corresponding subcase for the corner (m, m). This completes the proof. □

Recently, we improved on the upper bound for s_n given by Theorem 4.1. Using a similar but more complex indexing function, we showed that $s_n \leq 0.467n + \Theta(1)$. The details of the construction, which are tedious and do not provide any new insight into the problem, are not included in this paper. They can be found in [3].

5. Concluding remarks

In this paper we investigated parameter s_n that arises when proving lower bounds for sorting on a mesh-connected computer using the Chain Theorem. We showed that $0.27 + \Theta(1) \leq s_n \leq 0.5n + \Theta(1)$. The gap between the bounds is still quite big and leaves room for improvement. Improving the lower bound on s_n (we believe that it can be improved) would give a better lower bound for sorting on a mesh-connected computer. However, even the bound we were able to obtain indicates that, unlike in the case of 1-dimensional mesh-connected computer, in the 2-dimensional case the simple distance-based lower bound cannot be achieved. The upper bound on s_n, which we proved by exhibiting a class of indexing functions I with $s(I) = 0.5n + \Theta(1)$ shows the limit of the Chain Theorem in proving lower bounds. Although the Chain Theorem is strong enough to prove optimality (up to the leading term) of the $3n + o(n)$ algorithm of Schnorr and Schamir [15], it seems unlikely that an indexing function exists that would admit a sorting algorithm running in $(3 - \varepsilon)n + o(n)$ steps. So, in general, stronger lower-bound techniques are needed.

Another interesting problem is to improve the upper bounds for various indexing schemes. Even for the row-major indexing scheme we do not know whether there exists an algorithm sorting in less than $4n$ steps (the lower bound following from the Chain Theorem is $3n + \Theta(1)$). This problem seems particularly worth of future studies.

References

[1] Y. Han and Y. Igarashi, Time lower bounds for sorting on a mesh-connected processor array, Acta Inform. 26 (1989) 643–655.

[2] Y. Han and Y. Igarashi, Time lower bounds for parallel sorting on multidimensional mesh-connected processor arrays, Inform. Process. Lett. 33 (1990) 233–238.

[3] Y. Han, Y. Igarashi and M. Truszczynski, A new limit for the power of the Chain Theorem, Tech. Rept. 163-90, Department of Computer Science, University of Kentucky, Lexington, KY (1990).

[4] M. Kunde, Lower bounds for sorting on mesh-connected architectures, Acta Inform. 24 (1987) 121–130.

[5] M. Kunde, Optimal sorting on multi-dimensionally mesh-connected computers, in: Proceedings 4th

Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Computer Science 247 (Springer, Berlin, 1987) 408-419.

[6] M. Kunde, Bounds for 1-selection and related problems on grids of processors, in: Proceedings of the Fourth International Workshop on Parallel Processing by Cellular Automata and Arrays, Berlin, Lecture Notes in Computer Science 342 (Springer, Berlin, 1988) 298-307.

[7] H.-W. Lang, M. Schimmler, H. Schmech and H. Schröder, Systolic sorting on a mesh-connected network, IEEE Trans. Comput. 34 (1985) 652-658.

[8] Y. Ma, S. Sen and I.D. Scherson, The distance bound for sorting on mesh-connected processor arrays is tight, in: Proceedings 27th Symposium on Foundations of Computer Science (IEEE Press, New York, 1986) 255-263.

[9] D. Nassimi and S. Sahni, Bitonic sort on a mesh-connected parallel computer, IEEE Trans. Comput. 27 (1979) 2-7.

[10] K. Sado and Y. Igarashi, A divide-and-conquer method of the parallel sort, IECE of Japan, AL84-68, Tech. Commit. of Automata and Languages (1984) 41-50.

[11] K. Sado and Y. Igarashi, A fast pseudo-merge sorting algorithm, IECE of Japan, AL85-16, Tech. Commit. of Automata and Languages (1985) 21-30.

[12] K. Sado and Y. Igarashi, Fast parallel sorting on a mesh-connected processor array, in: D.S. Johnson et al., eds., Proceedings of Japan-U.S. Joint Seminar, Discrete Algorithms and Complexity Theory (Academic Press, New York, 1986) 161-183.

[13] K. Sado and Y. Igarashi, Some parallel sorts on a mesh-connected processor array, J. Parallel Distributed Comput. 3 (1986) 389-410.

[14] I.D. Scherson, S. Sen and A. Shamir, Shear sort: A true two-dimensional sorting technique for VLSI networks, in: Proceedings 1986 International Conference on Parallel Processing (1986) 903-908.

[15] C.P. Schnorr and A. Shamir, An optimal sorting algorithm for mesh-connected computers, in: Proceedings 18th ACM Symposium on Theory of Computing (1986) 255-263.

[16] C.D. Thompson and H.T. Kung, Sorting on a mesh-connected parallel computer, Comm. ACM 20 (1977) 263-271.